ATLORIA

Comparing Version Changes for Release Decisions

August 25, 2026

Opening the version comparison view for two releases

To compare releases in Atloria, start from the **Releases** or version list inside your project workspace. This guide assumes you already know how to reach version records and monitor generation results from Generating Documentation Versions and Monitoring Results. Before you begin, make sure you can open the release record for both versions you want to review.

In the comparison view, teams usually work with two selections:

- a **source release**, which acts as the baseline
- a **target release**, which is the candidate being reviewed for approval or publishing

In most release reviews, the **source release** is the last approved or published version, and the **target release** is the newer draft or generated version you want to evaluate. This makes it easier to answer the most important question: **what will change for readers if this release goes live?**

Look for the release or version picker at the top of the comparison screen. Use it to choose the older release first, then the newer release. If your team compares the wrong direction, the results can still show changes, but the meaning of added and removed items may feel reversed during review.

When choosing releases, pay attention to status labels shown in the version list or release record, such as:

- **Draft**
- **Published**
- **Archived**

A common review pattern is comparing a **Draft** candidate against the current **Published** release. Archived releases may still be useful as historical baselines, but they can introduce extra differences if they are not the most recent approved version.

[SCREENSHOT: version comparison screen with source release and target release selectors highlighted]

Reading what changed between the selected versions

Once both releases are selected, Atloria shows a comparison view that helps you scan changes quickly before opening any page in detail. The main comparison area typically combines a results list with a content comparison panel so you can move from high-level review to page-level inspection without leaving the screen.

Start with the changed-items list. This list groups documentation entries that differ between the two selected releases. Depending on your project structure, you may see changed pages, articles, navigation-linked content, or other documentation items. Each row in the list should help you identify the item by its title and related location information.

Use the status labels in the results to understand the scope of the release:

- **Added** means the page or content item exists in the target release but not in the source release
- **Updated** means the item exists in both releases, but its content or details changed
- **Removed** means the item existed in the source release and no longer appears in the target release
- **Unchanged** helps confirm that an item was reviewed but did not change

When you select an item, the comparison panel shows the difference in a side-by-side or diff-style layout. Look for visual highlights that mark inserted text, deleted text, and edited sections. This is especially useful for checking whether a page was lightly edited or substantially rewritten.

Atloria may also surface metadata changes alongside body content changes. Watch for updates to fields such as:

- **Title**
- **Slug** or page path
- **Last updated**
- **Publish state**

These details matter because a release can be affected by more than rewritten text. A renamed page, changed path, or different publish state can alter navigation, links, and reader access even when the body content looks similar.

[SCREENSHOT: comparison results list showing Added, Updated, Removed, and Unchanged labels]

Evaluating release readiness from the comparison results

The comparison screen is one of the fastest ways to decide whether a candidate release is complete enough for approval. Instead of reviewing every page from scratch, use the changed-items list to confirm that the release includes the updates your team expected to ship.

Begin by checking whether all planned documentation changes appear in the target release. If a feature launch was supposed to include a new getting started page, updated release notes, and revised setup instructions, each of those items should appear in the comparison results. Missing expected changes are often the first sign that the release is not ready.

Pay close attention to three warning patterns:

- **Missing removals** — old pages still appear even though they were supposed to be retired
- **Unexpected additions** — new pages or entries appear that were not part of the release scope
- **Incomplete edits** — a page shows as updated, but the actual content still looks partial or outdated

Documentation Managers often review high-impact pages first because these pages shape the reader experience immediately. Prioritize items such as:

- landing pages
- navigation entries
- release notes
- setup or onboarding pages
- pages linked from public navigation

If these areas are wrong, the release can feel incomplete even when most supporting pages are correct.

Technical Writers usually go one level deeper. They open updated pages and verify that the wording, structure, and coverage match the intended release scope. A page marked **Updated** is not automatically ready; it still needs to reflect the actual change set the team planned to publish.

If the comparison shows the right pages, the right type of changes, and no obvious gaps, the release is usually in a strong position for formal review. For broader status and lifecycle context, pair this screen with Comparing Documentation Versions for Release Decisions.

Using filters and detail views to investigate specific changes

When a release contains many changed pages, filtering becomes essential. Atloria's comparison view is most useful when you narrow the results to the type of work you need to review instead of scanning the full list repeatedly.

Use the change-type filters to isolate items by status, such as:

- **Added**
- **Modified** or **Updated**
- **Removed**

This is especially helpful during focused review sessions. For example, if you are validating a cleanup release, filter to **Removed** first to confirm outdated pages were actually taken out. If you are checking a feature launch, filter to **Added** and **Updated** to review the new and revised pages together.

Search within the comparison results when you need to find a specific item quickly. Search is useful for locating:

- a page title
- a path or slug
- a known content entry under review

After finding the item, open it from the results list to inspect the detailed differences. The detail view is where you can tell whether a change is minor editing or something more significant. For example, a small wording adjustment may only affect a few lines, while a structural change may alter headings, navigation labels, or page organization.

If Atloria offers both a summary view and a detailed view, switch between them based on the review question:

- use **summary** to understand release scope quickly
- use **detailed view** to inspect exact wording and field-level changes

This distinction matters during approval. Editorial fixes may be acceptable late in the process, while structural changes, path changes, or major content removals may require another review round before publishing.

[SCREENSHOT: filtered comparison results with a search box and a selected page showing detailed differences]

Supporting approval and publishing decisions with comparison data

During release sign-off, the comparison view gives reviewers a shared, concrete view of what will change if the candidate version is published. Instead of relying on memory or scattered comments, the team can point to the exact pages and content differences shown in Atloria.

Reviewers typically use the comparison screen to answer a few practical questions:

- Does this release include all required documentation updates?
- Are any unexpected pages being added or removed?
- Do the changed pages match the approved release scope?
- Are the changes small enough to approve now, or large enough to require more revision?

This makes approval conversations more precise. A reviewer can open the candidate release, compare it against the current published release, and show exactly which landing pages, release notes, or setup pages changed. That evidence helps teams decide whether to move forward or pause.

In general, teams tend to **approve immediately** when the comparison shows expected changes only, especially if high-impact pages are complete and the edits match the planned release. They may **request revisions** when the comparison reveals incomplete text, missing pages, accidental removals, or metadata changes that could break navigation. They may **delay publishing** when the release contains a large volume of unrelated edits or when the baseline comparison suggests the wrong release was used.

Comparison results also support traceability. When reviewers discuss why a release was approved, rejected, or delayed, the comparison output serves as a record of what changed at that decision point. For teams with formal review practices, this works well alongside version review and approval workflows described in Managing Version Review Decisions and Approvals and Preparing Versions for Final Approval.

Resolving common issues when comparing releases

If the comparison view does not look right, the problem is often caused by the selected releases, active filters, or access limits rather than missing content. Start by checking the controls at the top of the comparison screen before assuming the release itself is wrong.

If the comparison shows **no differences**, verify that the correct source and target releases are selected in the version picker. Teams sometimes compare a release against itself or choose two nearly identical drafts by mistake. Also confirm that you are comparing the intended baseline, usually the last approved or published release, against the candidate release.

If an **expected page is missing** from the results, check two things:

- whether the page belongs to the selected release
- whether a filter is hiding it

For example, if the results are filtered to **Removed**, you will not see pages that were only updated. Clear or adjust the change-type filter and search for the page title again.

If you see **too many unrelated changes**, the baseline release may be too old. Comparing a current draft against an older archived release can make the candidate look much larger than it really is. Switch the source release to the most recent approved or published version to get a cleaner review.

If a reviewer **cannot open comparison details**, the issue may be access-related. They may be able to see the release list but not open the changed content itself. In that case, confirm they have permission to view the release record and the related documentation entries.

When comparison results still seem inconsistent, go back to the version list and confirm the status and identity of each release before restarting the review.

Overview

Atloria’s release comparison view is designed to help teams make publishing decisions based on visible, reviewable changes between two versions. Instead of checking pages one by one across separate tabs, you can compare a baseline release and a candidate release in one place and quickly see what was added, updated, removed, or left unchanged.

The comparison workflow is most useful when you are close to approval and need confidence that the candidate release matches the intended scope. In practice, teams use it to:

- compare the current **Published** release with a newer **Draft**
- confirm that expected pages appear in the release
- spot unexpected additions or removals
- inspect detailed content differences before sign-off
- support approval and publishing discussions with clear evidence

The most effective review pattern is to start broad and then narrow your focus:

- review the overall changed-items list
- scan status labels such as **Added**, **Updated**, and **Removed**
- prioritize high-impact pages like navigation entries and release notes
- open detailed differences for any item that could affect release readiness

This document focuses on the comparison stage only. If you need help generating versions or tracking generation progress before review, use Managing Version Generation Jobs and Results and Generating Documentation Versions and Monitoring Results. If you need broader guidance on version status and comparison workflows, see Working with Version Comparison Views.

Used well, the comparison screen becomes the final checkpoint between a generated release and a publishing decision.

Prerequisites

Before comparing version changes in Atloria, make sure you already have access to the project workspace and can open the relevant release records. This task works best when the release has already been generated and is ready for review rather than still being created.

You should have the following in place:

- access to the project's **Releases** or version list
- at least two releases available to compare
- permission to open release records and view documentation content
- a clear understanding of which release is the current baseline and which is the candidate for approval

In most teams, the baseline is the latest approved or published release, while the candidate is a newer draft under review. If you are unsure which versions to choose, confirm that before starting the comparison. Picking the wrong baseline can make the results misleading and create unnecessary review work.

It also helps to know the intended release scope before opening the comparison view. Reviewers are much faster when they already know which pages, release notes, or navigation updates are expected. Bring that context into the comparison so you can quickly spot:

- missing pages
- unexpected removals
- unrelated edits
- incomplete updates

If you have not yet generated the candidate version or checked its processing results, complete that work first using Generating New Documentation Versions and Monitoring Version Generation Progress and Results.

After the comparison is complete and the release looks correct, the next step is usually formal review, approval, or publishing preparation. For that stage, continue with Managing Version Review Requests and Decisions.